PYTHON! Build Delta Layers

Contributed by Kevin Bell
07, Dec. 2007
Last Updated 10, Dec. 2007

The "Feature Compare" tool in ArcToolbox/Data Comparison can give you some unexpected results, so beware. It compares features row by row, and while you can specify a sort field, like your primary key, it will sort that field and then simply compare row by row, regardless of whether there are the same number of rows, or if primary keys have changed. The results look something like this: PK1
ATTRIBUTE1
PK2
ATTRIBUTE2
TOOLRESPONSE
KevinsRESPONSE
1 foo 1 foo same OK 2 bar 2 BARF different OK 3 x 3 x same OK 4 y 5 s different BAD 5 s 6 f different BAD 6 f 7 v different BAD
AD

So while Feature Class 1 & 2 are sorted on the PK, missing one row from FC2 (PK 4) causes a mismatch all the way through the rest of the rows. This may not be a problem for nearly identical data, but what if you want to track assets, comparing feature classes that have additions or deletions of features? This tools won't compare your data.

The python code below compares features based on a field that you specify, not sequentially row by row. Read the comments, and note that all of the data specific elements are hardcoded, so you'll have to change the feature classes to compare, the workspace, primary key and field names. If you'd like to tighten this up, you could dynamically build up the cursor body using gp.ListFields, etc. Send me the results!

Also, When people refer to python as an elegant language, you might ask "how so?" Look at the one-liner in the __valuesChanged function. This function is given 2 dictionaries, with KEY/VALUE pairs, in this case your primary key associated with all of is geometry definition and attributes. The single line of code compares the dictionaries for discrepencies and returns a list of primary keys with geometry or attribute changes. My friend Jonathon Ellis from the Utah Python Users Group helped out with that line! Pretty cool stuff if your a geek, eh? That one-liner is referred to as a list comprehension. Google it. Just in case you're wondering, the functions that are preceeded by a double underscore are private. The double underscore doesn't really do anything in this case other than remind you that these functions are called from other code, never directly.

Let it SNOW!!!

```
#NAME:  buildDeltaLYRs.py
#AUTHOR:  Kevin Bell
#EMAIL:  kevin.bell@slcgov.com
#DATE:  20071207

#PURPOSE:  create adds/deletes layer files by comparing 2 point
#         feature classes shape and attributes.  If the shape has
#         not changed, but any of the attributes have, the feature
#         will show as a delete, and an add.

#RECOMMENDED SYMBOL: adds.lyr = green plus, deletes.lyr = red X
#                   (this allows for nice stacking)

#NOTE:  __buildDict method has hard coded primary key and attribute names.



#XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

def buildDeltaLayers(inFC1, inFC2):
    '''build an adds and deletes lyr for a given chrono FC '''
    d1 = __buildDict(inFC1)
    d2 = __buildDict(inFC2)
    compareList = __valuesChanged(d1,d2)
    __makeLYR(inFC1, compareList, "deletes")
```

```
    print "...deletes.lyr is complete."

    d1 = __buildDict(inFC2)
    d2 = __buildDict(inFC1)
    compareList = __valuesChanged(d1,d2)
    __makeLYR(inFC2, compareList, "adds")
    print "...adds.lyr is complete."

def __valuesChanged(dict1, dict2):
    '''get a list of keys from one dict if a cooresponding dict's values are different '''
    outList = [key for key in set(dict1.keys() + dict2.keys()) if dict1.get(key) != dict2.get(key)]
    return outList

def __buildDict(inputFC): #-----BEWARE OF HARDCODED PRIMARY KEY AND ATTRIBUTES BELOW!!!!!
    '''Build a dictionary of the primary key, and it's fields'''
    d = {}
    cur = gp.SearchCursor(inputFC)
    row = cur.Next()
    while row:
        d[row.FP] = [row.Shape.Centroid, row.LUMENS, row.WATTS, row.TYPE, row.OWNER, row.RATE]
        row = cur.Next()
    del cur
    return d

def __makeLYR(fc, inList, outLyrName):# BEWARE OF HARDCODED PRIMARY KEY BELOW
    '''given a list, return a LYR file'''
    wc = str(tuple(inList))
    whereclause = "FP IN " + wc # <----IF DATA ISN'T FILE GDB, YOU MAY NEED QUOTES/BRACKETS
    gp.MakeFeatureLayer_management (fc, outLyrName, whereclause)
    gp.SaveToLayerFile_management  (outLyrName, outLyrName +".lyr")

#XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

print "-----------  building adds/deletes layer files  ----------------"

import arcgisscripting, time
startTime = time.clock()

gp = arcgisscripting.create()
gp.workspace = r"F:\Temp\streetLightCompare.gdb"# <----BEWARE OF HARDCODED WORKSPACE !!!!!
gp.OverwriteOutput = 1

#o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o
buildDeltaLayers("SEP07","OCT07") #<-------------------BEWARE OF HARDCODED FEATURE CLASSES !!!!!
#o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o

print "Your adds/deletes lyr's are in:"
print str(gp.workspace)

del gp

print "------------------------------------------------------------------------------------------"
stopTime = time.clock()
elapsedTime = stopTime - startTime
print "elapsed time = " + str(round(elapsedTime, 1)) + " seconds"
```